Published in williambastidasblog

William Bastidas Follow

Aug 2, 2020 · 2 min read

Angular decoradores @Input y @Output

Esta será una publicación rápida para mostrar cómo trabajar con los decoradores @Input y @Output en Angular.

Los decoradores @Input y @Output sirven para intercambiar datos entre componentes, son un mecanismo para enviar y recibir datos de un componente a otro.

@Input se usa para recibir datos en un componente mientras que @Output se usa para enviar datos fuera de un componente. @Output envía datos exponiendo a los

padre debemos agregarle el decorador **@input** a la propiedad que deseamos controlar.

Ejemplo:

home.component.ts

```
import { component, Input, Output } from '@angular/core'

@Component ({
  selector: 'home-component',
  template: `
    <p>
      {{value}}
    </p>
  `
})export class HelloComponent {
  @Input () value: string;

}
```

Con esta instrucción angular espera recibir un valor de tipo string y lo almacena en la propiedad *value* .

```
@Input () value: string;
```

Opcionalmente podríamos renombrar el valor de la propiedad:

```
@Input('saludo') value: string;
```

Ahora podemos enviar el valor desde el componente padre, en este caso el app.component.ts

app.component.ts

```
})export class HelloComponent {

}
```

Decorador @output

Para emitir un valor desde el componente hijo (home.component) que pueda ser escuchado por el componente padre (app.component) utilizamos el decorador @output.

home.component.ts

```
import { component, Input, Output } from '@angular/core'

@Component ({
  selector: 'home-component',
  template: `
    <p>
      {{value}}
    </p>
  `
})export class HelloComponent implements OnDestroy {

  @Input () value: string;
  @Output () valueResponse: EventEmitter<string> = new
EventEmitter();

}
```


Aunque no tiene mucho sentido pero para efectos de ejemplo, emitimos el valor de respuesta cuando se destruya el componente hijo (home), utilizando el ngOnDestroy.

home.component.ts

```
import { component, Input, Output } from '@angular/core'

@Component ({
  selector: 'home-component',
  template: `
```


```
@Input () value: string;
@Output () valueResponse: EventEmitter<string> = new EventEmitter();

ngOnDestroy() {
      this.valueResponse.emit("Bienvenido!!!");
}

}
```

Finalmente para obtener este evento en el componente padre escuchamos y definimos que hacer cuando se recibe ese evento de la siguiente manera:

app.component.ts

```
@Component ({
  selector: 'app-component',
  template: `
    <home-component (valueResponse)="recibiRespuesta($event)"
[value]="Hello World!" ></home-component>
  `
})export class HelloComponent {

  valueResponse(respuesta) {
   alert(respuesta);
  }

}
```

Conclusión

Con esto ya sabemos utilizar **input y output en Angular** para intercambiar datos entre componentes.

> Te ha gustado el Post? o crees que hay algo pueda mejorar o hacer de otra forma? no dudes en dejarlo en los comentarios!. Puedes estar en contacto conmigo en mis cuentas de Twitter, facebook y LinkedIn. ☺